

73305.P001

UNITED STATES PATENT APPLICATION FOR

INTEGRATED CIRCUIT I/O USING A HIGH
PERFORMANCE BUS INTERFACE

Inventor:

Michael Farmwald

Mark Horowitz

Prepared by:

BROWN & BAIN
600 Hansen Way
Suite 100
Palo Alto, California 94306
(415) 856-9411

I hereby certify that this correspondence is being
deposited with the United States Postal Service as
Express Mail (Label No. RB317101180) in an
envelope addressed to: Commissioner of Patents
and Trademarks, Washington, D.C. 20231.

on: April 18, 1990

(Date of Deposit)

Tricia Rivera 4/18/90
(Signature) (Reg. No.) (Date)

Our Ref: RamBus D-1

Integrated Circuit I/O Using A
High Performance Bus Interface

This application is a continuation of Ser. No. 08/798,520 filed Feb. 10, 1997, now Pat. No. 5,841,580.

FIELD OF THE INVENTION

An integrated circuit bus interface for computer and video systems is described which allows high speed transfer of blocks of data, particularly to and from memory devices, with reduced power consumption and increased system reliability. A new method of physically implementing the bus architecture is also described.

BACKGROUND OF THE INVENTION

Semiconductor computer memories have traditionally been designed and structured to use one memory device for each bit, or small group of bits, of any individual computer word, where the word size is governed by the choice of computer. Typical word sizes range from 4 to 64 bits. Each memory device typically is connected in parallel to a series of address lines and connected to one of a series of data lines. When the computer seeks to read from or write to a specific memory location, an address is put on the address lines and some or all of the memory devices are activated using a separate device select line for each needed device. One or more devices may be connected to each data line but typically only a small number of data lines are connected to

a single memory device. Thus data line 0 is connected to device(s) 0, data line 1 is connected to device(s) 1, and so on. Data is thus accessed or provided in parallel for each memory read or write operation. For the system to operate properly, every single memory bit in every memory device must operate dependably and correctly.

To understand the concept of the present invention, it is helpful to review the architecture of conventional memory devices. Internal to nearly all types of memory devices (including the most widely used Dynamic Random Access Memory (DRAM), Static RAM (SRAM) and Read Only Memory (ROM) devices), a large number of bits are accessed in parallel each time the system carries out a memory access cycle. However, only a small percentage of accessed bits which are available internally each time the memory device is cycled ever make it across the device boundary to the external world.

Referring to Fig. 1, all modern DRAM, SRAM and ROM designs have internal architectures with row (word) lines and column (bit) lines to allow the memory cells to tile a two dimensional area. One bit of data is stored at the intersection of each word and bit line. When a particular word line is enabled, all of the corresponding data bits are transferred onto the bit lines. Some prior art DRAMs take advantage of this organization to reduce the number of pins needed to transmit the address. The address of a given memory

cell is split into two addresses, row and column, each of which can be multiplexed over a bus only half as wide as the memory cell address of the prior art would have required.

5 COMPARISON WITH PRIOR ART

10 Prior art memory systems have attempted to solve the problem of high speed access to memory with limited success. U.S. Patent No. 3,821,715 (Hoff et. al.), was issued to Intel Corporation for the earliest 4-bit ^{MICROPROCESSOR} ~~micro-processor~~. That patent describes a bus connecting a single central processing unit (CPU) with multiple RAMs and ROMs. That bus multiplexes addresses and data over a 4-bit wide bus and uses point-to-point control signals to select particular RAMs or ROMs. The access time is fixed and only a single processing element is permitted. There is no block-mode type of operation, and most important, not all of the interface signals between the devices are bused (the ROM and RAM control lines and the RAM select lines are point-to-point).

15 In U.S. Patent No. 4,315,308 (Jackson), a bus connecting a single CPU to a bus interface unit is described. The invention uses multiplexed address, data, and control information over a single 16-bit wide bus. Block-mode operations are defined, with the length of the block sent as part of the control sequence. In addition, variable access-time operations using a "stretch" cycle signal are provided. There are no

multiple processing elements and no capability for multiple outstanding requests, and again, not all of the interface signals are bused.

5 In U.S. Patent No. 4,449,207 (Kung, et. al.), a DRAM is described which multiplexes address and data on an internal bus. The external interface to this DRAM is conventional, with separate control, address and data connections.

10 In U.S. Patent Nos. 4,764,846 and 4,706,166 (Go), a 3-D package arrangement of stacked die with connections along a single edge is described. Such packages are difficult to use because of the point-to-point wiring required to interconnect conventional memory devices with processing elements. Both patents describe complex schemes for solving these problems. No attempt is made to solve the problem by changing the interface.

15 In U.S. Patent No. 3,969,706 (Proebsting, et. al.), the current state-of-the-art DRAM interface is described. The address is two-way multiplexed, and there are separate pins for data and control (RAS, CAS, WE, CS). The number of pins grows with the size of the DRAM, and many of the connections must be made point-to-point in a memory system using such DRAMs.

20 There are many backplane buses described in the prior art, but not in the combination described or having the features of this invention. Many backplane buses multiplex addresses and data on a single bus (e.g., the NU bus). ELXSI and others have implemented split-transaction buses (U.S. Patent No. 4,595,923

and 4,481,625 (Roberts)). ELXSI has also implemented a relatively low-voltage-swing current-mode ECL driver (approximately 1 V swing). Address-space registers are implemented on most backplane buses, as is some form of block mode operation.

Nearly all modern backplane buses implement some type of arbitration scheme, but the arbitration scheme used in this invention differs from each of these. U.S. Patent Nos. 4,837,682 (Culler), 4,818,985 (Ikeda), 4,779,089 (Theus) and 4,745,548 (Blahut) describe prior art schemes. All involve either log N extra signals, (Theus, Blahut), where N is the number of potential bus requestors, or additional delay to get control of the bus (Ikeda, Culler). None of the buses described in patents or other literature use only bused connections. All contain some point-to-point connections on the backplane. None of the other aspects of this invention such as power reduction by fetching each data block from a single device or compact and low-cost 3-D packaging even apply to backplane buses.

The clocking scheme used in this invention has not been used before and in fact would be difficult to implement in backplane buses due to the signal degradation caused by connector stubs. U.S. Patent No. 4,247,817 (Heller) describes a clocking scheme using two clock lines, but relies on ramp-shaped clock signals in contrast to the normal rise-time signals used in the present invention.

4646,270

A
In U.S. Patent No. ~~4,646,279~~ (Voss), a video RAM is described which implements a parallel-load, serial-out shift register on the output of a DRAM. This generally allows greatly improved bandwidth (and has been extended to 2, 4 and greater width shift-out paths.) The rest of the interfaces to the DRAM (RAS, CAS, multiplexed address, etc.) remain the same as for conventional DRAMS.

5
10
15
20
25
30
35
40
45
50
55
60
65
70
75
80
85
90
95
100
105
110
115
120
125
130
135
140
145
150
155
160
165
170
175
180
185
190
195
200
205
210
215
220
225
230
235
240
245
250
255
260
265
270
275
280
285
290
295
300
305
310
315
320
325
330
335
340
345
350
355
360
365
370
375
380
385
390
395
400
405
410
415
420
425
430
435
440
445
450
455
460
465
470
475
480
485
490
495
500
505
510
515
520
525
530
535
540
545
550
555
560
565
570
575
580
585
590
595
600
605
610
615
620
625
630
635
640
645
650
655
660
665
670
675
680
685
690
695
700
705
710
715
720
725
730
735
740
745
750
755
760
765
770
775
780
785
790
795
800
805
810
815
820
825
830
835
840
845
850
855
860
865
870
875
880
885
890
895
900
905
910
915
920
925
930
935
940
945
950
955
960
965
970
975
980
985
990
995
1000
1005
1010
1015
1020
1025
1030
1035
1040
1045
1050
1055
1060
1065
1070
1075
1080
1085
1090
1095
1100
1105
1110
1115
1120
1125
1130
1135
1140
1145
1150
1155
1160
1165
1170
1175
1180
1185
1190
1195
1200
1205
1210
1215
1220
1225
1230
1235
1240
1245
1250
1255
1260
1265
1270
1275
1280
1285
1290
1295
1300
1305
1310
1315
1320
1325
1330
1335
1340
1345
1350
1355
1360
1365
1370
1375
1380
1385
1390
1395
1400
1405
1410
1415
1420
1425
1430
1435
1440
1445
1450
1455
1460
1465
1470
1475
1480
1485
1490
1495
1500
1505
1510
1515
1520
1525
1530
1535
1540
1545
1550
1555
1560
1565
1570
1575
1580
1585
1590
1595
1600
1605
1610
1615
1620
1625
1630
1635
1640
1645
1650
1655
1660
1665
1670
1675
1680
1685
1690
1695
1700
1705
1710
1715
1720
1725
1730
1735
1740
1745
1750
1755
1760
1765
1770
1775
1780
1785
1790
1795
1800
1805
1810
1815
1820
1825
1830
1835
1840
1845
1850
1855
1860
1865
1870
1875
1880
1885
1890
1895
1900
1905
1910
1915
1920
1925
1930
1935
1940
1945
1950
1955
1960
1965
1970
1975
1980
1985
1990
1995
2000
2005
2010
2015
2020
2025
2030
2035
2040
2045
2050
2055
2060
2065
2070
2075
2080
2085
2090
2095
2100
2105
2110
2115
2120
2125
2130
2135
2140
2145
2150
2155
2160
2165
2170
2175
2180
2185
2190
2195
2200
2205
2210
2215
2220
2225
2230
2235
2240
2245
2250
2255
2260
2265
2270
2275
2280
2285
2290
2295
2300
2305
2310
2315
2320
2325
2330
2335
2340
2345
2350
2355
2360
2365
2370
2375
2380
2385
2390
2395
2400
2405
2410
2415
2420
2425
2430
2435
2440
2445
2450
2455
2460
2465
2470
2475
2480
2485
2490
2495
2500
2505
2510
2515
2520
2525
2530
2535
2540
2545
2550
2555
2560
2565
2570
2575
2580
2585
2590
2595
2600
2605
2610
2615
2620
2625
2630
2635
2640
2645
2650
2655
2660
2665
2670
2675
2680
2685
2690
2695
2700
2705
2710
2715
2720
2725
2730
2735
2740
2745
2750
2755
2760
2765
2770
2775
2780
2785
2790
2795
2800
2805
2810
2815
2820
2825
2830
2835
2840
2845
2850
2855
2860
2865
2870
2875
2880
2885
2890
2895
2900
2905
2910
2915
2920
2925
2930
2935
2940
2945
2950
2955
2960
2965
2970
2975
2980
2985
2990
2995
3000
3005
3010
3015
3020
3025
3030
3035
3040
3045
3050
3055
3060
3065
3070
3075
3080
3085
3090
3095
3100
3105
3110
3115
3120
3125
3130
3135
3140
3145
3150
3155
3160
3165
3170
3175
3180
3185
3190
3195
3200
3205
3210
3215
3220
3225
3230
3235
3240
3245
3250
3255
3260
3265
3270
3275
3280
3285
3290
3295
3300
3305
3310
3315
3320
3325
3330
3335
3340
3345
3350
3355
3360
3365
3370
3375
3380
3385
3390
3395
3400
3405
3410
3415
3420
3425
3430
3435
3440
3445
3450
3455
3460
3465
3470
3475
3480
3485
3490
3495
3500
3505
3510
3515
3520
3525
3530
3535
3540
3545
3550
3555
3560
3565
3570
3575
3580
3585
3590
3595
3600
3605
3610
3615
3620
3625
3630
3635
3640
3645
3650
3655
3660
3665
3670
3675
3680
3685
3690
3695
3700
3705
3710
3715
3720
3725
3730
3735
3740
3745
3750
3755
3760
3765
3770
3775
3780
3785
3790
3795
3800
3805
3810
3815
3820
3825
3830
3835
3840
3845
3850
3855
3860
3865
3870
3875
3880
3885
3890
3895
3900
3905
3910
3915
3920
3925
3930
3935
3940
3945
3950
3955
3960
3965
3970
3975
3980
3985
3990
3995
4000
4005
4010
4015
4020
4025
4030
4035
4040
4045
4050
4055
4060
4065
4070
4075
4080
4085
4090
4095
4100
4105
4110
4115
4120
4125
4130
4135
4140
4145
4150
4155
4160
4165
4170
4175
4180
4185
4190
4195
4200
4205
4210
4215
4220
4225
4230
4235
4240
4245
4250
4255
4260
4265
4270
4275
4280
4285
4290
4295
4300
4305
4310
4315
4320
4325
4330
4335
4340
4345
4350
4355
4360
4365
4370
4375
4380
4385
4390
4395
4400
4405
4410
4415
4420
4425
4430
4435
4440
4445
4450
4455
4460
4465
4470
4475
4480
4485
4490
4495
4500
4505
4510
4515
4520
4525
4530
4535
4540
4545
4550
4555
4560
4565
4570
4575
4580
4585
4590
4595
4600
4605
4610
4615
4620
4625
4630
4635
4640
4645
4650
4655
4660
4665
4670
4675
4680
4685
4690
4695
4700
4705
4710
4715
4720
4725
4730
4735
4740
4745
4750
4755
4760
4765
4770
4775
4780
4785
4790
4795
4800
4805
4810
4815
4820
4825
4830
4835
4840
4845
4850
4855
4860
4865
4870
4875
4880
4885
4890
4895
4900
4905
4910
4915
4920
4925
4930
4935
4940
4945
4950
4955
4960
4965
4970
4975
4980
4985
4990
4995
5000
5005
5010
5015
5020
5025
5030
5035
5040
5045
5050
5055
5060
5065
5070
5075
5080
5085
5090
5095
5100
5105
5110
5115
5120
5125
5130
5135
5140
5145
5150
5155
5160
5165
5170
5175
5180
5185
5190
5195
5200
5205
5210
5215
5220
5225
5230
5235
5240
5245
5250
5255
5260
5265
5270
5275
5280
5285
5290
5295
5300
5305
5310
5315
5320
5325
5330
5335
5340
5345
5350
5355
5360
5365
5370
5375
5380
5385
5390
5395
5400
5405
5410
5415
5420
5425
5430
5435
5440
5445
5450
5455
5460
5465
5470
5475
5480
5485
5490
5495
5500
5505
5510
5515
5520
5525
5530
5535
5540
5545
5550
5555
5560
5565
5570
5575
5580
5585
5590
5595
5600
5605
5610
5615
5620
5625
5630
5635
5640
5645
5650
5655
5660
5665
5670
5675
5680
5685
5690
5695
5700
5705
5710
5715
5720
5725
5730
5735
5740
5745
5750
5755
5760
5765
5770
5775
5780
5785
5790
5795
5800
5805
5810
5815
5820
5825
5830
5835
5840
5845
5850
5855
5860
5865
5870
5875
5880
5885
5890
5895
5900
5905
5910
5915
5920
5925
5930
5935
5940
5945
5950
5955
5960
5965
5970
5975
5980
5985
5990
5995
6000
6005
6010
6015
6020
6025
6030
6035
6040
6045
6050
6055
6060
6065
6070
6075
6080
6085
6090
6095
6100
6105
6110
6115
6120
6125
6130
6135
6140
6145
6150
6155
6160
6165
6170
6175
6180
6185
6190
6195
6200
6205
6210
6215
6220
6225
6230
6235
6240
6245
6250
6255
6260
6265
6270
6275
6280
6285
6290
6295
6300
6305
6310
6315
6320
6325
6330
6335
6340
6345
6350
6355
6360
6365
6370
6375
6380
6385
6390
6395
6400
6405
6410
6415
6420
6425
6430
6435
6440
6445
6450
6455
6460
6465
6470
6475
6480
6485
6490
6495
6500
6505
6510
6515
6520
6525
6530
6535
6540
6545
6550
6555
6560
6565
6570
6575
6580
6585
6590
6595
6600
6605
6610
6615
6620
6625
6630
6635
6640
6645
6650
6655
6660
6665
6670
6675
6680
6685
6690
6695
6700
6705
6710
6715
6720
6725
6730
6735
6740
6745
6750
6755
6760
6765
6770
6775
6780
6785
6790
6795
6800
6805
6810
6815
6820
6825
6830
6835
6840
6845
6850
6855
6860
6865
6870
6875
6880
6885
6890
6895
6900
6905
6910
6915
6920
6925
6930
6935
6940
6945
6950
6955
6960
6965
6970
6975
6980
6985
6990
6995
7000
7005
7010
7015
7020
7025
7030
7035
7040
7045
7050
7055
7060
7065
7070
7075
7080
7085
7090
7095
7100
7105
7110
7115
7120
7125
7130
7135
7140
7145
7150
7155
7160
7165
7170
7175
7180
7185
7190
7195
7200
7205
7210
7215
7220
7225
7230
7235
7240
7245
7250
7255
7260
7265
7270
7275
7280
7285
7290
7295
7300
7305
7310
7315
7320
7325
7330
7335
7340
7345
7350
7355
7360
7365
7370
7375
7380
7385
7390
7395
7400
7405
7410
7415
7420
7425
7430
7435
7440
7445
7450
7455
7460
7465
7470
7475
7480
7485
7490
7495
7500
7505
7510
7515
7520
7525
7530
7535
7540
7545
7550
7555
7560
7565
7570
7575
7580
7585
7590
7595
7600
7605
7610
7615
7620
7625
7630
7635
7640
7645
7650
7655
7660
7665
7670
7675
7680
7685
7690
7695
7700
7705
7710
7715
7720
7725
7730
7735
7740
7745
7750
7755
7760
7765
7770
7775
7780
7785
7790
7795
7800
7805
7810
7815
7820
7825
7830
7835
7840
7845
7850
7855
7860
7865
7870
7875
7880
7885
7890
7895
7900
7905
7910
7915
7920
7925
7930
7935
7940
7945
7950
7955
7960
7965
7970
7975
7980
7985
7990
7995
8000
8005
8010
8015
8020
8025
8030
8035
8040
8045
8050
8055
8060
8065
8070
8075
8080
8085
8090
8095
8100
8105
8110
8115
8120
8125
8130
8135
8140
8145
8150
8155
8160
8165
8170
8175
8180
8185
8190
8195
8200
8205
8210
8215
8220
8225
8230
8235
8240
8245
8250
8255
8260
8265
8270
8275
8280
8285
8290
8295
8300
8305
8310
8315
8320
8325
8330
8335
8340
8345
8350
8355
8360
8365
8370
8375
8380
8385
8390
8395
8400
8405
8410
8415
8420
8425
8430
8435
8440
8445
8450
8455
8460
8465
8470
8475
8480
8485
8490
8495
8500
8505
8510
8515
8520
8525
8530
8535
8540
8545
8550
8555
8560
8565
8570
8575
8580
8585
8590
8595
8600
8605
8610
8615
8620
8625
8630
8635
8640
8645
8650
8655
8660
8665
8670
8675
8680
8685
8690
8695
8700
8705
8710
8715
8720
8725
8730
8735
8740
8745
8750
8755
8760
8765
8770
8775
8780
8785
8790
8795
8800
8805
8810
8815
8820
8825
8830
8835
8840
8845
8850
8855
8860
8865
8870
8875
8880
8885
8890
8895
8900
8905
8910
8915
8920
8925
8930
8935
8940
8945
8950
8955
8960
8965
8970
8975
8980
8985
8990
8995
9000
9005
9010
9015
9020
9025
9030
9035
9040
9045
9050
9055
9060
9065
9070
9075
9080
9085
9090
9095
9100
9105
9110
9115
9120
9125
9130
9135
9140
9145
9150
9155
9160
9165
9170
9175
9180
9185
9190
9195
9200
9205
9210
9215
9220
9225
9230
9235
9240
9245
9250
9255
9260
9265
9270
9275
9280
9285
9290
9295
9300
9305
9310
9315
9320
9325
9330
9335
9340
9345
9350
9355
9360
9365
9370
9375
9380
9385
9390
9395
9400
9405
9410
9415
9420
9425
9430
9435
9440
9445
9450
9455
9460
9465
9470
9475
9480
9485
9490
9495
9500
9505
9510
9515
9520
9525
9530
9535
9540
9545
9550
9555
9560
9565
9570
9575
9580
9585
9590
9595
9600
9605
9610
9615
9620
9625
9630
9635
9640
9645
9650
9655
9660
9665
9670
9675
9680
9685
9690
9695
9700
9705
9710
9715
9720
9725
9730
9735
9740
9745
9750
9755
9760
9765
9770
9775
9780
9785
9790
9795
9800
9805
9810
9815
9820
9825
9830
9835
9840
9845
9850
9855
9860
9865
9870
9875
9880
9885
9890
9895
9900
9905
9910
9915
9920
9925
9930
9935
9940
9945
9950
9955
9960
9965
9970
9975
9980
9985
9990
9995
10000
10005
10010
10015
10020
10025
10030
10035
10040
10045
10050
10055
10060
10065
10070
10075
10080
10085
10090
10095
10100
10105
10110
10115
10120
10125
10130
10135
10140
10145
10150
10155
10160
10165
10170
10175
10180
10185
10190
10195
10200
10205
10210
10215
10220
10225
10230
10235
10240
10245
10250
10255
10260
10265
10270
10275
10280
10285

Another object of this invention is to provide a method of distributing a high-speed memory cache within the DRAM chips of a memory system which is much more effective than previous cache methods.

5 Another object of this invention is to provide devices, especially DRAMs, suitable for use with the bus architecture of the invention.

SUMMARY OF INVENTION

10 The present invention includes a memory subsystem comprising at least two semiconductor devices, including at least one memory device, connected in parallel to a bus, where the bus includes a plurality of bus lines for carrying substantially all address, data and control information needed by said memory
15 devices, where the control information includes device-select information and the bus has substantially fewer bus lines than the number of bits in a single address, and the bus carries device-select information without the need for separate device-select lines connected directly to individual devices.

20 Referring to Fig. 2, a standard DRAM 13, 14, ROM (or SRAM) 12, microprocessor CPU 11, I/O device, disk controller or other special purpose device such as a high speed switch is modified to use a wholly bus-based interface rather than the
25 prior art combination of point-to-point and bus-based wiring used with conventional versions of these devices. The new bus

includes clock signals, power and multiplexed address, data and control signals. In a preferred implementation, 8 bus data lines and an AddressValid bus line carry address, data and control information for memory addresses up to 40 bits wide. Persons
5 skilled in the art will recognize that 16 bus data lines or other numbers of bus data lines can be used to implement the teaching of this invention. The new bus is used to connect elements such as memory, peripheral, switch and processing units.

10 In the system of this invention, DRAMs and other devices receive address and control information over the bus and transmit or receive requested data over the same bus. Each memory device contains only a single bus interface with no other signal pins. Other devices that may be included in the system can connect to the bus and other non-bus lines, such as
15 input/output lines. The bus supports large data block transfers and split transactions to allow a user to achieve high bus utilization. This ability to rapidly read or write a large block of data to one single device at a time is an important advantage of this invention.

20 The DRAMs that connect to this bus differ from conventional DRAMs in a number of ways. Registers are provided which may store control information, device identification, device-type and other information appropriate for the chip such as the address range for each independent portion of the device.

25 New bus interface circuits must be added and the internals of

prior art DRAM devices need to be modified so they can provide and accept data to and from the bus at the peak data rate of the bus. This requires changes to the column access circuitry in the DRAM, with only a minimal increase in die size. A circuit is provided to generate a low skew internal device clock for devices on the bus, and other circuits provide for demultiplexing input and multiplexing output signals.

High bus bandwidth is achieved by running the bus at a very high clock rate (hundreds of MHz). This high clock rate is made possible by the constrained environment of the bus. The bus lines are controlled-impedance, doubly-terminated lines. For a data rate of 500 MHz, the maximum bus propagation time is less than 1 ns (the physical bus length is about 10 cm). In addition, because of the packaging used, the pitch of the pins can be very close to the pitch of the pads. The loading on the bus resulting from the individual devices is very small. In a preferred implementation, this generally allows stub capacitances of 1-2 pF and inductances of 0.5 - 2 nH. Each device 15, 16, 17, shown in Figure 3, only has pins on one side and these pins connect directly to the bus 18. A transceiver device 19 can be included to interface multiple units to a higher order bus through pins 20.

A primary result of the architecture of this invention is to increase the bandwidth of DRAM access. The invention also

reduces manufacturing and production costs, power consumption, and increases packing density and system reliability.

BRIEF DESCRIPTION OF THE DRAWINGS

5 Figure 1 is a diagram which illustrates the basic 2-D organization of memory devices.

 Figure 2 is a schematic block diagram which illustrates the parallel connection of all bus lines and the serial Reset line to each device in the system.

10 Figure 3 is a perspective view of a system of the invention which illustrates the 3-D packaging of semiconductor devices on the primary bus.

 Figure 4 shows the format of a request packet.

15 Figure 5 shows the format of a retry response from a slave.

 Figure 6 shows the bus cycles after a request packet collision occurs on the bus and how arbitration is handled.

20 *Figures 7a and 7b show*
 ~~Figure 7~~ shows the timing whereby signals from two devices can overlap temporarily and drive the bus at the same time.

Figures 8a and 8b show
 ~~Figure 8~~ shows the connection and timing between bus clocks and devices on the bus.

 Figure 9 is a perspective view showing how transceivers can be used to connect a number of bus units to a transceiver

bus. Figure 10 is a block and schematic diagram of input/output circuitry used to connect devices to the bus.

Figure 11 is a schematic diagram of a clocked sense-amplifier used as a bus input receiver.

5 Figure 12 is a block diagram showing how the internal device clock is generated from two bus clock signals using a set of adjustable delay lines.

Figure 13 is a timing diagram showing the relationship of signals in the block diagram of Figure 12.

10 Figure 14 is timing diagram of a preferred means of implementing the reset procedure of this invention.

Figure 15 is a diagram illustrating the general organization of a 4 Mbit DRAM divided into 8 subarrays.

15 DETAILED DESCRIPTION

The present invention is designed to provide a high speed, multiplexed bus for communication between processing devices and memory devices and to provide devices adapted for use in the bus system. The invention can also be used to connect
20 processing devices and other devices, such as I/O interfaces or disk controllers, with or without memory devices on the bus. The bus consists of a relatively small number of lines connected in parallel to each device on the bus. The bus carries
substantially all address, data and control information needed by
25 devices for communication with other devices on the bus. In many

systems using the present invention, the bus carries almost every signal between every device in the entire system. There is no need for separate device-select lines since device-select information for each device on the bus is carried over the bus.

5 There is no need for separate address and data lines because address and data information can be sent over the same lines. Using the organization described herein, very large addresses (40 bits in the preferred implementation) and large data blocks (1024 bytes) can be sent over a small number of bus lines (8 plus one control line in the preferred implementation).

10 Virtually all of the signals needed by a computer system can be sent over the bus. Persons skilled in the art recognize that certain devices, such as CPUs, may be connected to other signal lines and possibly to independent buses, for example a bus to an independent cache memory, in addition to the bus of this invention. Certain devices, for example cross-point switches, could be connected to multiple, independent buses of this invention. In the preferred implementation, memory devices are provided that have no connections other than the bus connections described herein and CPUs are provided that use the bus of this invention as the principal, if not exclusive, connection to memory and to other devices on the bus.

15 All modern DRAM, SRAM and ROM designs have internal architectures with row (word) and column (bit) lines to efficiently tile a 2-D area. Referring to Fig. 1, one bit of

data is stored at the intersection of each word line 5 and bit line 6. When a particular word line is enabled, all of the corresponding data bits are transferred onto the bit lines. This data, about 4000 bits at a time in a 4 MBit DRAM, is then loaded into column sense amplifiers 3 and held for use by the I/O circuits.

10 In the invention presented here, the data from the sense amplifiers is enabled 32 bits at a time onto an internal device bus running at approximately 125 MHz. This internal device bus moves the data to the periphery of the devices where the data is multiplexed into an 8-bit wide external bus interface, running at approximately 500 MHz.

15 The bus architecture of this invention connects master or bus controller devices, such as CPUs, Direct Memory Access devices (DMAs) or Floating Point Units (FPUs), and slave devices, such as DRAM, SRAM or ROM memory devices. A slave device responds to control signals; a master sends control signals. Persons skilled in the art realize that some devices may behave as both master and slave at various times, depending on the mode of operation and the state of the system. For example, a memory device will typically have only slave functions, while a DMA controller, disk controller or CPU may include both slave and master functions. Many other semiconductor devices, including I/O devices, disk controllers, or other special purpose devices

such as high speed switches can be modified for use with the bus of this invention.

Emb 2 > Each semiconductor device contains a set of internal registers, preferably including a device identification (device ID) register, a device-type descriptor register, control registers, and other registers containing other information relevant to that type of device. In a preferred implementation, semiconductor devices connected to the bus contain registers which specify the memory addresses contained within that device and access-time registers which store a set of one or more delay times at which the device can or should be available to send or receive data.

Most of these registers can be modified and preferably are set as part of an initialization sequence that occurs when the system is powered up or reset. During the initialization sequence each device on the bus is assigned a unique device ID number, which is stored in the device ID register. A bus master can then use these device ID numbers to access and set appropriate registers in other devices, including access-time registers, control registers, and memory registers to configure the system. Each slave may have one or several access-time registers (four in a preferred embodiment). In a preferred embodiment, one access-time register in each slave is permanently or semi-permanently programmed with a fixed value to facilitate

certain control functions. A preferred implementation of an initialization sequence is described below in more detail.

10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100

All information sent between master devices and slave devices is sent over the external bus, which, for example, may be 8 bits wide. This is accomplished by defining a protocol whereby a master device, such as a microprocessor, seizes exclusive control of the external bus (i.e., becomes the bus master) and initiates a bus transaction by sending a request packet (a sequence of bytes comprising address and control information) to one or more slave devices on the bus. An address can consist of 16 to 40 or more bits according to the teachings of this invention. Each slave on the bus must decode the request packet to see if that slave needs to respond to the packet. The slave that the packet is directed to must then begin any internal processes needed to carry out the requested bus transaction at the requested time. The requesting master may also need to transact certain internal processes before the bus transaction begins. After a specified access time the slave(s) respond by returning one or more bytes (8 bits) of data or by storing information made available from the bus. More than one access time can be provided to allow different types of responses to occur at different times.

A request packet and the corresponding bus access are separated by a selected number of bus cycles, allowing the bus to be used in the intervening bus cycles by the same or other

masters for additional requests or brief bus accesses. Thus multiple, independent accesses are permitted, allowing maximum utilization of the bus for transfer of short blocks of data. Transfers of long blocks of data use the bus efficiently even without overlap because the overhead due to bus address, control and access times is small compared to the total time to request and transfer the block.

Device Address Mapping

Another unique aspect of this invention is that each memory device is a complete, independent memory subsystem with all the functionality of a prior art memory board in a conventional backplane-bus computer system. Individual memory devices may contain a single memory section or may be subdivided into more than one discrete memory section. Memory devices preferably include memory address registers for each discrete memory section. A failed memory device (or even a subsection of a device) can be "mapped out" with only the loss of a small fraction of the memory, maintaining essentially full system capability. Mapping out bad devices can be accomplished in two ways, both compatible with this invention.

The preferred method uses address registers in each memory device (or independent discrete portion thereof) to store information which defines the range of bus addresses to which this memory device will respond. This is similar to prior art

schemes used in memory boards in conventional backplane bus systems. The address registers can include a single pointer, usually pointing to a block of known size, a pointer and a fixed or variable block size value or two pointers, one pointing to the beginning and one to the end (or to the "top" and "bottom") of each memory block. By appropriate settings of the address registers, a series of functional memory devices or discrete memory sections can be made to respond to a contiguous range of addresses, giving the system access to a contiguous block of good memory, limited primarily by the number of good devices connected to the bus. A block of memory in a first memory device or memory section can be assigned a certain range of addresses, then a block of memory in a next memory device or memory section can be assigned addresses starting with an address one higher (or lower, depending on the memory structure) than the last address of the previous block.

Preferred devices for use in this invention include device-type register information specifying the type of chip, including how much memory is available in what configuration on that device. A master can perform an appropriate memory test, such as reading and writing each memory cell in one or more selected orders, to test proper functioning of each accessible discrete portion of memory (based in part on information like device ID number and device-type) and write address values (up to 40 bits in the preferred embodiment, 10^{12} bytes), preferably

contiguous, into device address-space registers. Non-functional or impaired memory sections can be assigned a special address value which the system can interpret to avoid using that memory.

The second approach puts the burden of avoiding the bad devices on the system master or masters. CPUs and DMA controllers typically have some sort of translation look-aside buffers (TLBs) which map virtual to physical (bus) addresses. With relatively simple software, the TLBs can be programmed to use only working memory (data structures describing functional memories are easily generated). For masters which don't contain TLBs (for example, a video display generator), a small, simple RAM can be used to map a contiguous range of addresses onto the addresses of the functional memory devices.

Either scheme works and permits a system to have a significant percentage of non-functional devices and still continue to operate with the memory which remains. This means that systems built with this invention will have much improved reliability over existing systems, including the ability to build systems with almost no field failures.

Bus

The preferred bus architecture of this invention comprises 11 signals: BusData[0:7]; AddrValid; Clk1 and Clk2; plus an input reference level and power and ground lines connected in parallel to each device. Signals are driven onto

the bus during conventional bus cycles. The notation "Signal[i:j]" refers to a specific range of signals or lines, for example, BusData[0:7] means BusData0, BusData1, . . ., BusData7. The bus lines for BusData[0:7] signals form a byte-wide, multiplexed data/address/control bus. AddrValid is used to indicate when the bus is holding a valid address request, and instructs a slave to decode the bus data as an address and, if the address is included on that slave, to handle the pending request. The two clocks together provide a synchronized, high speed clock for all the devices on the bus. In addition to the bused signals, there is one other line (ResetIn, ResetOut) connecting each device in series for use during initialization to assign every device in the system a unique device ID number (described below in detail).

To facilitate the extremely high data rate of this external bus relative to the gate delays of the internal logic, the bus cycles are grouped into pairs of even/odd cycles. Note that all devices connected to a bus should preferably use the same even/odd labeling of bus cycles and preferably should begin operations on even cycles. This is enforced by the clocking scheme.

Protocol and Bus Operation

The bus uses a relatively simple, synchronous, split-transaction, block-oriented protocol for bus transactions. One

of the goals of the system is to keep the intelligence concentrated in the masters, thus keeping the slaves as simple as possible (since there are typically many more slaves than masters). To reduce the complexity of the slaves, a slave should preferably respond to a request in a specified time, sufficient to allow the slave to begin or possibly complete a device-internal phase including any internal actions that must precede the subsequent bus access phase. The time for this bus access phase is known to all devices on the bus - each master being responsible for making sure that the bus will be free when the bus access begins. Thus the slaves never worry about arbitrating for the bus. This approach eliminates arbitration in single master systems, and also makes the slave-bus interface simpler.

In a preferred implementation of the invention, to initiate a bus transfer over the bus, a master sends out a request packet, a contiguous series of bytes containing address and control information. It is preferable to use a request packet containing an even number of bytes and also preferable to start each packet on an even bus cycle.

The device-select function is handled using the bus data lines. AddrValid is driven, which instructs all slaves to decode the request packet address, determine whether they contain the requested address, and if they do, provide the data back to the master (in the case of a read request) or accept data from the master (in the case of a write request) in a data block

transfer. A master can also select a specific device by transmitting a device ID number in a request packet. In a preferred implementation, a special device ID number is chosen to indicate that the packet should be interpreted by all devices on the bus. This allows a master to broadcast a message, for example to set a selected control register of all devices with the same value.

The data block transfer occurs later at a time specified in the request packet control information, preferably beginning on an even cycle. A device begins a data block transfer almost immediately with a device-internal phase as the device initiates certain functions, such as setting up memory addressing, before the bus access phase begins. The time after which a data block is driven onto the bus lines is selected from values stored in slave access-time registers. The timing of data for reads and writes is preferably the same; the only difference is which device drives the bus. For reads, the slave drives the bus and the master latches the values from the bus. For writes the master drives the bus and the selected slave latches the values from the bus.

In a preferred implementation of this invention shown in Figure 4, a request packet 22 contains 6 bytes of data -- 4.5 address bytes and 1.5 control bytes. Each request packet uses all nine bits of the multiplexed data/address lines (AddrValid 23 + BusData[0:7] 24) for all six bytes of the request packet.

Setting 23 AddrValid = 1 in an otherwise unused even cycle indicates the start of an request packet (control information). In a valid request packet, AddrValid 27 must be 0 in the last byte. Asserting this signal in the last byte invalidates the request packet. This is used for the collision detection and arbitration logic (described below). Bytes 25-26 contain the first 35 address bits, Address[0:35]. The last byte contains AddrValid 27 (the invalidation switch) and 28, the remaining address bits, Address[36:39], and BlockSize[0:3] (control information).

The first byte contains two 4 bit fields containing control information, AccessType[0:3], an op code (operation code) which, for example, specifies the type of access, and Master[0:3], a position reserved for the master sending the packet to include its master ID number. Only master numbers 1 through 15 are allowed - master number 0 is reserved for special system commands. Any packet with Master[0:3] = 0 is an invalid or special packet and is treated accordingly.

The AccessType field specifies whether the requested operation is a read or write and the type of access, for example, whether it is to the control registers or other parts of the device, such as memory. In a preferred implementation, AccessType[0] is a Read/Write switch: if it is a 1, then the operation calls for a read from the slave (the slave to read the requested memory block and drive the memory contents onto the

bus); if it is a 0, the operation calls for a write into the slave (the slave to read data from the bus and write it to memory). AccessType[1:3] provides up to 8 different access types for a slave. AccessType[1:2] preferably indicates the timing of the response, which is stored in an access-time register, AccessRegN. The choice of access-time register can be selected directly by having a certain op code select that register, or indirectly by having a slave respond to selected op codes with pre-selected access times (see table below). The remaining bit, AccessType[3] may be used to send additional information about the request to the slaves.

One special type of access is control register access, which involves addressing a selected register in a selected slave. In the preferred implementation of this invention, AccessType[1:3] equal to zero indicates a control register request and the address field of the packet indicates the desired control register. For example, the most significant two bytes can be the device ID number (specifying which slave is being addressed) and the least significant three bytes can specify a register address and may also represent or include data to be loaded into that control register. Control register accesses are used to initialize the access-time registers, so it is preferable to use a fixed response time which can be preprogrammed or even hard wired, for example the value in AccessReg0, preferably 8

cycles. Control register access can also be used to initialize or modify other registers, including address registers.

The method of this invention provides for access mode control specifically for the DRAMs. One such access mode determines whether the access is page mode or normal RAS access. In normal mode (in conventional DRAMS and in this invention), the DRAM column sense amps or latches have been precharged to a value intermediate between logical 0 and 1. This precharging allows access to a row in the RAM to begin as soon as the access request for either inputs (writes) or outputs (reads) is received and allows the column sense amps to sense data quickly. In page mode (both conventional and in this invention), the DRAM holds the data in the column sense amps or latches from the previous read or write operation. If a subsequent request to access data is directed to the same row, the DRAM does not need to wait for the data to be sensed (it has been sensed already) and access time for this data is much shorter than the normal access time. Page mode generally allows much faster access to data but to a smaller block of data (equal to the number of sense amps). However, if the requested data is not in the selected row, the access time is longer than the normal access time, since the request must wait for the RAM to precharge before the normal mode access can start. Two access-time registers in each DRAM preferably contain the access times to be used for normal and for page-mode accesses, respectively.

5 The access mode also determines whether the DRAM should precharge the sense amplifiers or should save the contents of the sense amps for a subsequent page mode access. Typical settings are "precharge after normal access" and "save after page mode access" but "precharge after page mode access" or "save after normal access" are allowed, selectable modes of operation. The DRAM can also be set to precharge the sense amps if they are not accessed for a selected period of time.

10 In page mode, the data stored in the DRAM sense amplifiers may be accessed within much less time than it takes to read out data in normal mode (~10-20 nS vs. 40-100 nS). This data may be kept available for long periods. However, if these sense amps (and hence bit lines) are not precharged after an access, a subsequent access to a different memory word (row) will suffer a precharge time penalty of about 40-100 nS because the sense amps must precharge before latching in a new value.

15 The contents of the sense amps thus may be held and used as a cache, allowing faster, repetitive access to small blocks of data. DRAM-based page-mode caches have been attempted in the prior art using conventional DRAM organizations but they are not very effective because several chips are required per computer word. Such a conventional page-mode cache contains many bits (for example, 32 chips x 4Kbits) but has very few independent storage entries. In other words, at any given point in time the sense amps hold only a few different blocks or memory

"locales" (a single block of 4K words, in the example above).

Simulations have shown that upwards of 100 blocks are required to achieve high hit rates (>90% of requests find the requested data already in cache memory) regardless of the size of each block.

5 See, for example, Anant Agarwal, et. al., "An Analytic Cache Model," *ACM Transactions on Computer Systems*, Vol. 7(2), pp. 184-215 (May 1989).

10 The organization of memory in the present invention allows each DRAM to hold one or more (4 for 4MBit DRAMS) separately- addressed and independent blocks of data. A personal computer or workstation with 100 such DRAMS (i.e. 400 blocks or locales) can achieve extremely high, very repeatable hit rates (98-99% on average) as compared to the lower (50-80%), widely varying hit rates using DRAMS organized in the conventional
15 fashion. Further, because of the time penalty associated with the deferred precharge on a "miss" of the page-mode cache, the conventional DRAM-based page-mode cache generally has been found to work less well than no cache at all.

For DRAM slave access, the access types are preferably used in the following way:

<u>AccessType[1:3]</u>	<u>Use</u>	<u>AccessTime</u>
0	Control Register Access	Fixed, 8[AccessReg0]
1	Unused	Fixed, 8[AccessReg0]
2-3	Unused	AccessReg1
4-5	Page Mode DRAM access	AccessReg2
6-7	Normal DRAM access	AccessReg3

Persons skilled in the art will recognize that a series of available bits could be designated as switches for controlling these access modes. For example:

AccessType[2] = page mode/normal switch

AccessType[3] = precharge/save-data switch

BlockSize[0:3] specifies the size of the data block transfer. If BlockSize[0] is 0, the remaining bits are the binary representation of the block size (0-7). If BlockSize[0] is 1, then the remaining bits give the block size as a binary power of 2, from 8 to 1024. A zero-length block can be interpreted as a special command, for example, to refresh a DRAM without returning any data, or to change the DRAM from page mode to normal access mode or vice-versa.

T290X

BlockSize[0:2]

Number of Bytes in Block

0-7	0-7 respectively
8	8
9	16
10	32
11	64
12	128
13	256
14	512
15	1024

Persons skilled in the art will recognize that other block size encoding schemes or values can be used.

In most cases, a slave will respond at the selected access time by reading or writing data from or to the bus over bus lines BusData[0:7] and AddrValid will be at logical 0. In a preferred embodiment, substantially each memory access will involve only a single memory device, that is, a single block will be read from or written to a single memory device.

Retry Format

In some cases, a slave may not be able to respond correctly to a request, e.g., for a read or write. In such a situation, the slave should return an error message, sometimes called a N(o)ACK(nowledge) or retry message. The retry message can include information about the condition requiring a retry, but this increases system requirements for circuitry in both slave and masters. A simple message indicating only that an error has occurred allows for a less complex slave, and the

master can take whatever action is needed to understand and correct the cause of the error.

For example, under certain conditions a slave might not be able to supply the requested data. During a page-mode access, the DRAM selected must be in page mode and the requested address must match the address of the data held in the sense amps or latches. Each DRAM can check for this match during a page-mode access. If no match is found, the DRAM begins precharging and returns a retry message to the master during the first cycle of the data block (the rest of the returned block is ignored). The master then must wait for the precharge time (which is set to accommodate the type of slave in question, stored in a special register, PreChargeReg), and then resend the request as a normal DRAM access (AccessType = 6 or 7).

In the preferred form of the present invention, a slave signals a retry by driving AddrValid true at the time the slave was supposed to begin reading or writing data. A master which expected to write to that slave must monitor AddrValid during the write and take corrective action if it detects a retry message. Figure 5 illustrates the format of a retry message 28 which is useful for read requests, consisting of 23 AddrValid=1 with Master[0:3] = 0 in the first (even) cycle. Note that AddrValid is normally 0 for data block transfers and that there is no master 0 (only 1 through 15 are allowed). All DRAMs and masters can easily recognize such a packet as an invalid request packet,

and therefore a retry message. In this type of bus transaction all of the fields except for Master[0:3] and AddrValid 23 may be used as information fields, although in the implementation described, the contents are undefined. Persons skilled in the art recognize that another method of signifying a retry message is to add a DataInvalid line and signal to the bus. This signal could be asserted in the case of a NACK.

Bus Arbitration

In the case of a single master, there are by definition no arbitration problems. The master sends request packets and keeps track of periods when the bus will be busy in response to that packet. The master can schedule multiple requests so that the corresponding data block transfers do not overlap.

The bus architecture of this invention is also useful in configurations with multiple masters. When two or more masters are on the same bus, each master must keep track of all the pending transactions, so each master knows when it can send a request packet and access the corresponding data block transfer. Situations will arise, however, where two or more masters send a request packet at about the same time and the multiple requests must be detected, then sorted out by some sort of bus arbitration.

There are many ways for each master to keep track of when the bus is and will be busy. A simple method is for each

master to maintain a bus-busy data structure, for example by maintaining two pointers, one to indicate the earliest point in the future when the bus will be busy and the other to indicate the earliest point in the future when the bus will be free, that is, the end of the latest pending data block transfer. Using this information, each master can determine whether and when there is enough time to send a request packet (as described above under Protocol) before the bus becomes busy with another data block transfer and whether the corresponding data block transfer will interfere with pending bus transactions. Thus each master must read every request packet and update its bus-busy data structure to maintain information about when the bus is and will be free.

With two or more masters on the bus, masters will occasionally transmit independent request packets during the same bus cycle. Those multiple requests will collide as each such master drives the bus simultaneously with different information, resulting in scrambled request information and neither desired data block transfer. In a preferred form of the invention, each device on the bus seeking to write a logical 1 on a BusData or AddrValid line drives that line with a current sufficient to sustain a voltage greater than or equal to the high-logic value for the system. Devices do not drive lines that should have a logical 0; those lines are simply held at a voltage corresponding to a low-logic value. Each master tests the voltage on at least

some, preferably all, bus data and the AddrValid lines so the master can detect a logical '1' where the expected level is '0' on a line that it does not drive during a given bus cycle but another master does drive.

5 Another way to detect collisions is to select one or more bus lines for collision signalling. Each master sending a request drives that line or lines and monitors the selected lines for more than the normal drive current (or a logical value of
10 "1"), indicating requests by more than one master. Persons skilled in the art will recognize that this can be implemented with a protocol involving BusData and AddrValid lines or could be implemented using an additional bus line.

15 In the preferred form of this invention, each master detects collisions by monitoring lines which it does not drive to see if another master is driving those lines. Referring to Fig. 4, the first byte of the request packet includes the number of each master attempting to use the bus (Master[0:3]). If two masters send packet requests starting at the same point in time, the master numbers will be logical "or"ed together by at least
20 those masters, and thus one or both of the masters, by monitoring the data on the bus and comparing what it sent, can detect a collision. For instance if requests by masters number 2 (0010) and 5 (0101) collide, the bus will be driven with the value Master[0:3]=7 (0010 + 0101 = 0111). Master number 5 will detect
25 that the signal Master[2] = 1 and master 2 will detect that

Master[1] and Master[3] = 1, telling both masters that a collision has occurred. Another example is masters 2 and 11, for which the bus will be driven with the value Master[0:3]=11 (0010 + 1011 = 1011), and although master 11 can't readily detect this collision, master 2 can. When any collision is detected, each master detecting a collision drives the value of AddrValid 27 in byte 5 of the request packet 22 to 1, which is detected by all masters, including master 11 in the second example above, and forces a bus arbitration cycle, described below.

Another collision condition may arise where master A sends a request packet in cycle 0 and master B tries to send a request packet starting in cycle 2 of the first request packet, thereby overlapping the first request packet. This will occur from time to time because the bus operates at high speeds, thus the logic in a second-initiating master may not be fast enough to detect a request initiated by a first master in cycle 0 and to react fast enough by delaying its own request. Master B eventually notices that it wasn't supposed to try to send a request packet (and consequently almost surely destroyed the address that master A was trying to send), and, as in the example above of a simultaneous collision, drives a 1 on AddrValid during byte 5 of the first request packet 27 forcing an arbitration. The logic in the preferred implementation is fast enough that a master should detect a request packet by another master by cycle

3 of the first request packet, so no master is likely to attempt to send a potentially colliding request packet later than cycle 2.

Slave devices ^{do} not need to detect a collision directly, but they must wait to do anything irrecoverable until the last byte (byte 5) is read to ensure that the packet is valid. A request packet with Master[0:3] equal to 0 (a retry signal) is ignored and does not cause a collision. The subsequent bytes of such a packet are ignored.

To begin arbitration after a collision, the masters wait a preselected number of cycles after the aborted request packet (4 cycles in a preferred implementation), then use the next free cycle to arbitrate for the bus (the next available even cycle in the preferred implementation). Each colliding master signals to all other colliding masters that it seeks to send a request packet, a priority is assigned to each of the colliding masters, then each master is allowed to make its request in the order of that priority.

Figure 6 illustrates one preferred way of implementing this arbitration. Each colliding master signals its intent to send a request packet by driving a single BusData line during a single bus cycle corresponding to its assigned master number (1-15 in the present example). During two-byte arbitration cycle 29, byte 0 is allocated to requests 1-7 from masters 1-7, respectively, (bit 0 is not used) and byte 1 is allocated to

requests 8-15 from masters 8-15, respectively. At least one device and preferably each colliding master reads the values on the bus during the arbitration cycles to determine and store which masters desire to use the bus. Persons skilled in the art will recognize that a single byte can be allocated for arbitration requests if the system includes more bus lines than masters. More than 15 masters can be accommodated by using additional bus cycles.

A fixed priority scheme (preferably using the master numbers, selecting lowest numbers first) is then used to prioritize, then sequence the requests in a bus arbitration queue which is maintained by at least one device. These requests are queued by each master in the bus-busy data structure and no further requests are allowed until the bus arbitration queue is cleared. Persons skilled in the art will recognize that other priority schemes can be used, including assigning priority according to the physical location of each master.

System Configuration/Reset

In the bus-based system of this invention, a mechanism is provided to give each device on the bus a unique device identifier (device ID) after power-up or under other conditions as desired or needed by the system. A master can then use this device ID to access a specific device, particularly to set or modify registers¹⁷⁰ of the specified device, including the control

and address registers. In the preferred embodiment, one master is assigned to carry out the entire system configuration process. The master provides a series of unique device ID numbers for each unique device connected to the bus system. In the preferred embodiment, each device connected to the bus contains a special device-type register which specifies the type of device, for instance CPU, 4 MBit memory, 64 MBit memory or disk controller. The configuration master should check each device, determine the device type and set appropriate control registers, including access-time registers. The configuration master should check each memory device and set all appropriate memory address registers.

One means to set up unique device ID numbers is to have each device to select a device ID in sequence and store the value in an internal device ID register. For example, a master can pass sequential device ID numbers through shift registers in each of a series of devices, or pass a token from device to device whereby the device with the token reads in device ID information from another line or lines. In a preferred embodiment, device ID numbers are assigned to devices according to their physical relationship, for instance, their order along the bus.

In a preferred embodiment of this invention, the device ID setting is accomplished using a pair of pins on each device, ResetIn and ResetOut. These pins handle normal logic signals and are used only during device ID configuration. On each rising

edge of the clock, each device copies ResetIn (an input) into a four-stage reset shift register. The output of the reset shift register is connected to ResetOut, which in turn connects to ResetIn for the next sequentially connected device.

5 Substantially all devices on the bus are thereby daisy-chained together. A first reset signal, for example, while ResetIn at a device is a logical 1, or when a selected bit of the reset shift register goes from zero to non-zero, causes the device to hard
10 reset, for example by clearing all internal registers and resetting all state machines. A second reset signal, for example, the falling edge of ResetIn combined with changeable values on the external bus, causes that device to latch the contents of the external bus into the internal device ID register (Device[0:7]).

15 To reset all devices on a bus, a master sets the ResetIn line of the first device to a "1" for long enough to ensure that all devices on the bus have been reset (4 cycles times the number of devices -- note that the maximum number of devices on the preferred bus configuration is 256 (8 bits), so
20 that 1024 cycles is always enough time to reset all devices.) Then ResetIn is dropped to "0" and the BusData lines are driven with the first followed by successive device ID numbers, changing after every 4 clock pulses. Successive devices set those device ID numbers into the corresponding device ID register as the
25 falling edge of ResetIn propagates through the shift registers of

the daisy-chained devices. Figure 14 shows ResetIn at a first device going low while a master drives a first device ID onto the bus data lines BusData[0:3]. The first device then latches in that first device ID. After four clock cycles, the master changes BusData[0:3] to the next device ID number and ResetOut at the first device goes low, which pulls ResetIn for the next daisy-chained device low, allowing the next device to latch in the next device ID number from BusData[0:3]. In the preferred embodiment, one master is assigned device ID 0 and it is the responsibility of that master to control the ResetIn line and to drive successive device ID numbers onto the bus at the appropriate times. In the preferred embodiment, each device waits two clock cycles after ResetIn goes low before latching in a device ID number from BusData[0:3].

Persons skilled in the art recognize that longer device ID numbers could be distributed to devices by having each device read in multiple bytes from the bus and latch the values into the device ID register. Persons skilled in the art also recognize that there are alternative ways of getting device ID numbers to unique devices. For instance, a series of sequential numbers could be clocked along the ResetIn line and at a certain time each device could be instructed to latch the current reset shift register value into the device ID register.

The configuration master should choose and set an access time in each access-time register¹⁷³ in each slave to a

period sufficiently long to allow the slave to perform an actual, desired memory access. For example, for a normal DRAM access, this time must be longer than the row address strobe (RAS) access time. If this condition is not met, the slave may not deliver the correct data. The value stored in a slave access-time register¹⁷³ is preferably one-half the number of bus cycles for which the slave device should wait before using the bus in response to a request. Thus an access time value of '1' would indicate that the slave should not access the bus until at least two cycles after the last byte of the request packet has been received. The value of AccessReg0 is preferably fixed at 8 (cycles) to facilitate access to control registers.

The bus architecture of this invention can include more than one master device. The reset or initialization sequence should also include a determination of whether there are multiple masters on the bus, and if so to assign unique master ID numbers to each. Persons skilled in the art will recognize that there are many ways of doing this. For instance, the master could poll each device to determine what type of device it is, for example, by reading a special register then, for each master device, write the next available master ID number into a special register.

ECC

Error detection and correction ("ECC") methods well known in the art can be implemented in this system. ECC

information typically is calculated for a block of data at the time that block of data is first written into memory. The data block usually has an integral binary size, e.g. 256 bits, and the ECC information uses significantly fewer bits. A potential problem arises in that each binary data block in prior art schemes typically is stored with the ECC bits appended, resulting in a block size that is not an integral binary power.

In a preferred embodiment of this invention, ECC information is stored separately from the corresponding data, which can then be stored in blocks having integral binary size. ECC information and corresponding data can be stored, for example, in separate DRAM devices. Data can be read without ECC using a single request packet, but to write or read error-corrected data requires two request packets, one for the data and a second for the corresponding ECC information. ECC information may not always be stored permanently and in some situations the ECC information may be available without sending a request packet or without a bus data block transfer.

In a preferred embodiment, a standard data block size can be selected for use with ECC, and the ECC method will determine the required number of bits of information in a corresponding ECC block. RAMs containing ECC information can be programmed to store an access time that is equal to: (1) the access time of the normal RAM (containing data) plus the time to access a standard data block (for corrected data) minus the time

to send a request packet (6 bytes); ^{or} (2) the access time of a normal RAM minus the time to access a standard ECC block minus the time to send a request packet. To read a data block and the corresponding ECC block, the master simply issues a request for the data immediately followed by a request for the ECC block. The ECC RAM will wait for the selected access time then drive its data onto the bus right after (in case (1) above)) the data RAM has finished driving out the data block. Persons skilled in the art will recognize that the access time described in case (2) above can be used to drive ECC data before the data is driven onto the bus lines and will recognize that writing data can be done by analogy with the method described for a read. Persons skilled in the art will also recognize the adjustments that must be made in the bus-busy structure and the request packet arbitration methods of this invention in order to accommodate these paired ECC requests.

Since this system is quite flexible, the system designer can choose the size of the data blocks and the number of ECC bits using the memory devices of this invention. Note that the data stream on the bus can be interpreted in various ways. For instance the sequence can be 2^n data bytes followed by 2^n ECC bytes (or vice versa), or the sequence can be 2^k iterations of 8 data bytes plus 1 ECC byte. Other information, such as information used by a directory-based cache coherence scheme, can also be managed this way. See, for example, Anant Agarwal, et

al., "Scaleable Directory Schemes for Cache Consistency," 15th
International Symposium on Computer Architecture, June 1988, pp.
280-289. Those skilled in the art will recognize alternative
methods of implementing ECC schemes that are within the teachings
of this invention.

Low Power 3-D Packaging

Another major advantage of this invention is that it
drastically reduces the memory system power consumption. Nearly
all the power consumed by a prior art DRAM is dissipated in
performing row access. By using a single row access in a single
RAM to supply all the bits for a block request (compared to a
row-access in each of multiple RAMs in conventional memory
systems) the power per bit can be made very small. Since the
power dissipated by memory devices using this invention is
significantly reduced, the devices potentially can be placed much
closer together than with conventional designs.

The bus architecture of this invention makes possible
an innovative 3-D packaging technology. By using a narrow,
multiplexed (time-shared) bus, the pin count for an arbitrarily
large memory device can be kept quite small - on the order of 20
pins. Moreover, this pin count can be kept constant from one
generation of DRAM density to the next. The low power
dissipation allows each package to be smaller, with narrower pin
itches (spacing between the IC pins). With current surface

mount technology supporting pin pitches as low as 20 mils, all off-device connections can be implemented on a single edge of the memory device. Semiconductor die useful in this invention preferably have connections or pads along one edge of the die which can then be wired or otherwise connected to the package pins with wires having similar lengths. This geometry also allows for very short leads, preferably with an effective lead length of less than 4 mm. Furthermore, this invention uses only bused interconnections, i.e., each pad on each device is connected by the bus to the corresponding pad of each other device.

The use of a low pin count and an edge-connected bus permits a simple 3-D package, whereby the devices are stacked and the bus is connected along a single edge of the stack. The fact that all of the signals are bused is important for the implementation of a simple 3-D structure. Without this, the complexity of the "backplane" would be too difficult to make cost effectively with current technology. The individual devices in a stack of the present invention can be packed quite tightly because of the low power dissipated by the entire memory system, permitting the devices to be stacked bumper-to-bumper or top to bottom. Conventional plastic-injection molded small outline (SO) packages can be used with a pitch of about 2.5 mm (100 mils), but the ultimate limit would be the device die thickness, which is

about an order of magnitude smaller, 0.2-0.5 mm using current wafer technology.

Bus Electrical Description

5 By using devices with very low power dissipation and close physical packing, the bus can be made quite short, which in turn allows for short propagation times and high data rates. The bus of a preferred embodiment of the present invention consists of a set of resistor-terminated controlled impedance transmission
10 lines which can operate up to a data rate of 500 MHz (2 ns cycles). The characteristics of the transmission lines are strongly affected by the loading caused by the DRAMs (or other slaves) mounted on the bus. These devices add lumped capacitance to the lines which both lowers the impedance of the lines and
15 decreases the transmission speed. In the loaded environment, the bus impedance is likely to be on the order of 25 ohms and the propagation velocity about $c/4$ (c = the speed of light) or 7.5 cm/ns. To operate at a 2 ns data rate, the transit time on the bus should preferably be kept under 1 ns, to leave 1 ns for the
20 setup and hold time of the input receivers (described below) plus clock skew. Thus the bus lines must be kept quite short, under about 8 cm for maximum performance. Lower performance systems may have much longer lines, e.g. a 4 ns bus may have 24 cm lines (3 ns transit time, 1 ns setup and hold time).

In the preferred embodiment, the bus uses current source drivers. Each output must be able to sink 50 mA, which provides an output swing of about 500 mV or more. In the preferred embodiment of this invention, the bus is active low.

5 The unasserted state (the high value) is preferably considered a logical zero, and the asserted value (low state) is therefore a logical 1. Those skilled in the art understand that the method of this invention can also be implemented using the opposite logical relation to voltage. The value of the unasserted state is set by the voltage on the termination resistors, and should be high enough to allow the outputs to act as current sources, while being as low as possible to reduce power dissipation. These constraints may yield a termination voltage about 2V above ground in the preferred implementation. Current source drivers cause the output voltage to be proportional to the sum of the sources driving the bus.

Figures 7a and 7b
Referring to ~~Fig. 7~~, although there is no stable condition where two devices drive the bus at the same time, conditions can arise because of propagation delay on the wires where one device, A 41, can start driving its part of the bus 44 while the bus is still being driven by another device, B 42 (already asserting a logical 1 on the bus). In a system using current drivers, when B 42 is driving the bus (before time 46), the value at points 44 and 45 is logical 1. If B 42 switches off at time 46 just when A 41 switches on, the additional drive by

device A 41 causes the voltage at the output 44 of A 41 to drop briefly below the normal value. The voltage returns to its normal value at time 47 when the effect of device B 42 turning off is felt. The voltage at point 45 goes to logical 0 when device B 42 turns off, then drops at time 47 when the effect of device A 41 turning on is felt. Since the logical 1 driven by current from device A 41 is propagated irrespective of the previous value on the bus, the value on the bus is guaranteed to settle after one time of flight (t_f) delay, that is, the time it takes a signal to propagate from one end of the bus to the other. If a voltage drive was used (as in ECL wired-ORing), a logical 1 on the bus (from device B 42 being previously driven) would prevent the transition put out by device A 41 being felt at the most remote part of the system, e.g., device 43, until the turnoff waveform from device B 42 reached device A 41 plus one time of flight delay, giving a worst case settling time of twice the time of flight delay.

Clocking

Clocking a high speed bus accurately without introducing error due to propagation delays can be implemented by having each device monitor two bus clock signals and then derive internally a device clock, the true system clock. The bus clock information can be sent on one or two lines to provide a mechanism for each bused device to generate an internal device

clock with zero skew relative to all the other device clocks.

Figures 8a
Referring to Figure 8, in the preferred implementation, a bus clock generator 50 at one end of the bus propagates an early bus clock signal in one direction along the bus, for example on line 53 ~~from left to right~~ *from right to left*, to the far end of the bus. The same clock signal then is passed through the direct connection shown to a second line 54, and returns as a late bus clock signal along the bus from the far end to the origin, propagating from ~~right to left~~ *left to right*. A single bus clock line can be used if it is left unterminated at the far end of the bus, allowing the early bus clock signal to reflect back along the same line as a late bus clock signal.

Figure 8b illustrates how each device 51, 52 receives each of the two bus clock signals at a different time (because of propagation delay along the wires), with constant midpoint in time between the two bus clocks along the bus. At each device 51, 52, the rising edge 55 of Clock1 53 is followed by the rising edge 56 of Clock2 54. Similarly, the falling edge 57 of Clock1 53 is followed by the falling edge 58 of Clock2 54. This waveform relationship is observed at all other devices along the bus. Devices which are closer to the clock generator have a greater separation between Clock1 and Clock2 relative to devices farther from the generator because of the longer time required for each clock pulse to traverse the bus and return along line 54, but the midpoint in time 59, 60 between corresponding rising

or falling edges is fixed because, for any given device, the length of each clock line between the far end of the bus and that device is equal. Each device must sample the two bus clocks and generate its own internal device clock at the midpoint of the two.

Clock distribution problems can be further reduced by using a bus clock and device clock rate equal to the bus cycle data rate divided by two, that is, the bus clock period is twice the bus cycle period. Thus a 500 MHz bus preferably uses a 250 MHz clock rate. This reduction in frequency provides two benefits. First it makes all signals on the bus have the same worst case data rates -- data on a 500 MHz bus can only change every 2 ns. Second, clocking at half the bus cycle data rate makes the labeling of the odd and even bus cycles trivial, for example, by defining even cycles to be those when the internal device clock is 0 and odd cycles when the internal device clock is 1.

Multiple Buses

The limitation on bus length described above restricts the total number of devices that can be placed on a single bus. Using 2.5 mm spacing between devices, a single 8 cm bus will hold about 32 devices. Persons skilled in the art will recognize certain applications of the present invention wherein the overall data rate on the bus is adequate but memory or processing

requirements necessitate a much larger number of devices (many more than 32). Larger systems can easily be built using the teachings of this invention by using one or more memory subsystems, designated primary bus units, each of which consists of two or more devices, typically 32 or close to the maximum allowed by bus design requirements, connected to a transceiver device.

Referring to Figure 9, each primary bus unit can be mounted on a single circuit board 66, sometimes called a memory stick. Each transceiver device 19 in turn connects to a transceiver bus 65, similar or identical in electrical and other respects to the primary bus 18 described at length above. In a preferred implementation, all masters are situated on the transceiver bus so there are no transceiver delays between masters and all memory devices are on primary bus units so that all memory accesses experience an equivalent transceiver delay, but persons skilled in the art will recognize how to implement systems which have masters on more than one bus unit and memory devices on the transceiver bus as well as on primary bus units. In general, each teaching of this invention which refers to a memory device can be practiced using a transceiver device and one or more memory devices on an attached ^{primary} bus unit. Other devices, generically referred to as peripheral devices, including disk controllers, video controllers or I/O devices can also be attached to either the transceiver bus or a primary bus unit, as

desired. Persons skilled in the art will recognize how to use a single primary bus unit or multiple primary bus units as needed with a transceiver bus in certain system designs.

The transceivers are quite simple in function. They detect request packets on the transceiver bus and transmit them to their primary bus unit. If the request packet calls for a write to a device on a transceiver's primary bus unit, that transceiver keeps track of the access time and block size and forwards all data from the transceiver bus to the primary bus unit during that time. The transceivers also watch their primary bus unit, forwarding any data that occurs there to the transceiver bus. The high speed of the buses means that the transceivers will need to be pipelined, and will require an additional one or two cycle delay for data to pass through the transceiver in either direction. Access times stored in masters on the transceiver bus must be increased to account for transceiver delay but access times stored in slaves on a primary bus unit should not be modified.

Persons skilled in the art will recognize that a more sophisticated transceiver can control transmissions to and from primary bus units. An additional control line, TrncvrRW can be bused to all devices on the transceiver bus, using that line in conjunction with the AddrValid line to indicate to all devices on the transceiver bus that the information on the data lines is: 1) a request packet, 2) valid data to a slave, 3) valid data from a

slave, or 4) invalid data (or idle bus). Using this extra control line obviates the need for the transceivers to keep track of when data needs to be forwarded from its primary bus to the transceiver bus - all transceivers send all data from their primary bus to the transceiver bus whenever the control signal indicates condition 2) above. In a preferred implementation of this invention, if AddrValid and TrncvrRW are both low, there is no bus activity and the transceivers should remain in an idle state. A controller sending a request packet will drive AddrValid high, indicating to all devices on the transceiver bus that a request packet is being sent which each transceiver should forward to its primary bus unit. Each controller seeking to write to a slave should drive both AddrValid and TrncvrRW high, indicating valid data for a slave is present on the data lines. Each transceiver device will then transmit all data from the transceiver bus lines to each primary bus unit. Any controller expecting to receive information from a slave should also drive the TrncvrRW line high, but not drive AddrValid, thereby indicating to each transceiver to transmit any data coming from any slave on its primary local bus to the transceiver bus. A still more sophisticated transceiver would recognize signals addressed to or coming from its primary bus unit and transmit signals only at requested times.

An example of the physical mounting of the transceivers is shown in Figure 9. One important feature of this physical

arrangement is to integrate the bus of each transceiver 19 with the original bus of DRAMs or other devices 15, 16, 17 on the primary bus unit 66. The transceivers 19 have pins on two sides, and are preferably mounted flat on the primary bus unit with a first set of pins connected to primary bus 18. A second set of transceiver pins 20, preferably orthogonal to the first set of pins, are oriented to allow the transceiver 19 to be attached to the transceiver bus 65 in much the same way as the DRAMs were attached to the primary bus unit. The transceiver bus can be generally planar and in a different plane, preferably orthogonal to the plane of each primary bus unit. The transceiver bus can also be generally circular with primary bus units mounted perpendicular and tangential to the transceiver bus.

Using this two level scheme allows one to easily build a system that contains over 500 slaves (16 buses of 32 DRAMs each). Persons skilled in the art can modify the device ID scheme described above to accommodate more than 256 devices, for example by using a longer device ID or by using additional registers to hold some of the device ID. This scheme can be extended in yet a third dimension to make a second-order transceiver bus, connecting multiple transceiver buses by aligning transceiver bus units parallel to and on top of each other and busing corresponding signal lines through a suitable transceiver. Using such a second-order transceiver bus, one

could connect many thousands of slave devices into what is effectively a single bus.

Device Interface

5 The device interface to the high-speed bus can be divided into three main parts. The first part is the electrical interface. This part includes the input receivers, bus drivers and clock generation circuitry. The second part contains the address comparison circuitry and timing registers. This part
10 takes the input request packet and determines if the request is for this device, and if it is, starts the internal access and delivers the data to the pins at the correct time. The final part, specifically for memory devices such as DRAMs, is the DRAM column access path. This part needs to provide bandwidth into
15 and out of the DRAM sense amps greater than the bandwidth provided by conventional DRAMs. The implementation of the electrical interface and DRAM column access path are described in more detail in the following sections. Persons skilled in the art recognize how to modify prior-art address comparison
20 circuitry and prior-art register circuitry in order to practice the present invention.

Electrical Interface - Input/Output Circuitry

25 A block diagram of the preferred input/output circuit for address/data/control lines is shown in Figure 10. This

circuitry is particularly well-suited for use in DRAM devices but it can be used or modified by one skilled in the art for use in other devices connected to the bus of this invention. It

consists of a set of input receivers 71, 72 and output driver 76 connected to input/output line 69 and pad 75 and circuitry to use the internal clock 73 and internal clock complement 74 to drive the input interface. The clocked input receivers take advantage of the synchronous nature of the bus. To further reduce the performance requirements for device input receivers, each device pin, and thus each bus line, is connected to two clocked receivers, one to sample the even cycle inputs, the other to sample the odd cycle inputs. By thus de-multiplexing the input ⁶⁹ at the pin, each clocked amplifier is given a full 2 ns cycle to amplify the bus low-voltage-swing signal into a full value CMOS logic signal. Persons skilled in the art will recognize that additional clocked input receivers can be used within the teachings of this invention. For example, four input receivers could be connected to each device pin and clocked by a modified internal device clock to transfer sequential bits from the bus to internal device circuits, allowing still higher external bus speeds or still longer settling times to amplify the bus low-voltage-swing signal into a full value CMOS logic signal.

The output drivers are quite simple, and consist of a single NMOS pulldown transistor 76. This transistor is sized so that under worst case conditions it can still sink the 50 mA

required by the bus. For 0.8 micron CMOS technology, the transistor will need to be about 200 microns long. Overall bus performance can be improved by using feedback techniques to control output transistor current so that the current through the device is roughly 50 mA under all operating conditions, although this is not absolutely necessary for proper bus operation. An example of one of many methods known to persons skilled in the art for using feedback techniques to control current is described in Hans Schumacher, et al., "CMOS Subnanosecond True-ECL Output Buffer," *J. Solid State Circuits*, Vol. 25 (1), pp. 150-154 (Feb. 1990). Controlling this current improves performance and reduces power dissipation. This output driver which can be operated at 500 MHz, can in turn be controlled by a suitable multiplexer with two or more (preferably four) inputs connected to other internal chip circuitry, all of which can be designed according to well known prior art.

The input receivers of every slave must be able to operate during every cycle to determine whether the signal on the bus is a valid request packet. This requirement leads to a number of constraints on the input circuitry. In addition to requiring small acquisition and resolution delays, the circuits must take little or no DC power, little AC power and inject very little current back into the input or reference lines. The standard clocked DRAM sense amp shown in Figure 11 satisfies all these requirements except the need for low input currents. When

A this sense amp goes from sense to sample, the capacitance of the internal nodes 83 and 84 in ~~Figure 11~~ ^{Figure 11} is discharged through the reference line 68 and input 69, respectively. This particular current is small, but the sum of such currents from all the inputs into the reference lines summed over all devices can be reasonably large.

10 The fact that the sign of the current depends upon on the previous received data makes matters worse. One way to solve this problem is to divide the sample period into two phases. During the first phase, the inputs are shorted to a buffered version of the reference level (which may have an offset). During the second phase, the inputs are connected to the true inputs. This scheme does not remove the input current completely, since the input must still charge nodes 83 and 84 from the reference value to the current input value, but it does reduce the total charge required by about a factor of 10 (requiring only a 0.25V change rather than a 2.5V change). Persons skilled in the art will recognize that many other methods can be used to provide a clocked amplifier that will operate on very low input currents.

25 One important part of the input/output circuitry generates an internal device clock based on early and late bus clocks. Controlling clock skew (the difference in clock timing between devices) is important in a system running with 2 ns cycles, thus the internal device clock is generated so the input

sampler and the output driver operate as close in time as possible to midway between the two bus clocks.

A block diagram of the internal device clock generating circuit is shown in Figure 12 and the corresponding timing

5 diagram in Figure 13. The basic idea behind this circuit is relatively simple. A DC amplifier 102 is used to convert the small-swing bus clock into a full-swing CMOS signal. This signal is then fed into a variable delay line 103. The output of delay line 103 feeds three additional delay lines: 104 having a fixed delay; 105 having the same fixed delay plus a second variable delay; and 106 having the same fixed delay plus one half of the second variable delay. The outputs 107, 108 of the delay lines 104 and 105 drive clocked input receivers 101 and 111 connected to early and late bus clock inputs 100 and 110, respectively.

10 These input receivers 101 and 111 have the same design as the receivers described above and shown in Fig. 11. Variable delay lines 103 and 105 are adjusted via feedback lines 116, 115 so that input receivers 101 and 111 sample the bus clocks just as they transition. Delay lines 103 and 105 are adjusted so that
15 the falling edge 120 of output 107 precedes the falling edge 121 of the early bus clock, Clock1 53, by an amount of time 128 equal to the delay in input sampler 101. Delay line 108 is adjusted in the same way so that falling edge 122 precedes the falling edge 123 of late bus clock, Clock2 54, by the delay 128 in input
20 sampler 111.

Since the outputs 107 and 108 are synchronized with the two bus clocks and the output 73 of the last delay line 106 is midway between outputs 107 and 108, that is, output 73 follows output 107 by the same amount of time 129 that output 73 precedes output 108, output 73 provides an internal device clock midway between the bus clocks. The falling edge 124 of internal device clock 73 precedes the time of actual input sampling 125 by one sampler delay. Note that this circuit organization automatically balances the delay in substantially all device input receivers 71 and 72 (Fig. 10), since outputs 107 and 108 are adjusted so the bus clocks are sampled by input receivers 101 and 111 just as the bus clocks transition.

In the preferred embodiment, two sets of these delay lines are used, one to generate the true value of the internal device clock 73, and the other to generate the complement 74 without adding any inverter delay. The dual circuit allows generation of truly complementary clocks, with extremely small skew. The complement internal device clock is used to clock the 'even' input receivers to sample at time 127, while the true internal device clock is used to clock the 'odd' input receivers to sample at time 125. The true and complement internal device clocks are also used to select which data is driven to the output drivers. The gate delay between the internal device clock and output circuits driving the bus is slightly greater than the corresponding delay for the input circuits, which means that the

new data always will be driven on the bus slightly after the old data has been sampled.

DRAM Column Access Modification

5 A block diagram of a conventional 4 MBit DRAM 130 is shown in Figure 15. The DRAM memory array is divided into a number of subarrays 150-157, for example, 8. Each subarray is divided into arrays 148, 149 of memory cells. Row address selection is performed by decoders 146. A column decoder 147A, 147B, including column sense amps on either side of the decoder, runs through the core of each subarray. These column sense amps can be set to precharge or latch the most-recently stored value, as described in detail above. Internal I/O lines connect each set of sense-amps, as gated by corresponding column decoders, to input and output circuitry connected ultimately to the device pins. These internal I/O lines are used to drive the data from the selected bit lines to the data pins (some of pins 131-145), or to take the data from the pins and write the selected bit lines. Such a column access path organized by prior art constraints does not have sufficient bandwidth to interface with a high speed bus. The method of this invention does not require changing the overall method used for column access, but does change implementation details. Many of these details have been implemented selectively in certain fast memory devices, but never in conjunction with the bus architecture of this invention.

Running the internal I/O lines in the conventional way at high bus cycle rates is not possible. In the preferred method, several (preferably 4) bytes are read or written during each cycle and the column access path is modified to run at a lower rate (the inverse of the number of bytes accessed per cycle, preferably 1/4 of the bus cycle rate). Three different techniques are used to provide the additional internal I/O lines required and to supply data to memory cells at this rate. First, the number of I/O bit lines in each subarray running through the column decoder 147^{A,B} is increased, for example, to 16, eight for each of the two columns of column sense amps and the column decoder selects one set of columns from the "top" half 148 of subarray 150 and one set of columns from the "bottom" half 149 during each cycle, where the column decoder selects one column sense amp per I/O bit line. Second, each column I/O line is divided into two halves, carrying data independently over separate internal I/O lines from the left half 147A and right half 147B of each subarray (dividing each subarray into quadrants) and the column decoder selects sense amps from each right and left half of the subarray, doubling the number of bits available at each cycle. Thus each column decode selection turns on n column sense amps, where n equals four (top left and right, bottom left and right quadrants) times the number of I/O lines in the bus to each subarray quadrant (8 lines each $\times 4 = 32$ lines in the preferred implementation). Finally, during each RAS cycle,

two different subarrays, e.g. 157 and 153, are accessed. This doubles again the available number of I/O lines containing data. Taken together, these changes increase the internal I/O bandwidth by at least a factor of 8. Four internal buses are used to route these internal I/O lines. Increasing the number of I/O lines and then splitting them in the middle greatly reduces the capacitance of each internal I/O line which in turn reduces the column access time, increasing the column access bandwidth even further.

The multiple, gated input receivers described above allow high speed input from the device pins onto the internal I/O lines and ultimately into memory. The multiplexed output driver described above is used to keep up with the data flow available using these techniques. Control means are provided to select whether information at the device pins should be treated as an address, and therefore to be decoded, or input or output data to be driven onto or read from the internal I/O lines.

Each subarray can access 32 bits per cycle, 16 bits from the left subarray and 16 from the right subarray. With 8 I/O lines per sense-amplifier column and accessing two subarrays at a time, the DRAM can provide 64 bits per cycle. This extra I/O bandwidth is not needed for reads (and is probably not used), but may be needed for writes. Availability of write bandwidth is a more difficult problem than read bandwidth because over-writing a value in a sense-amplifier may be a slow operation, depending on how the sense amplifier is connected to the bit line. The

extra set of internal I/O lines provides some bandwidth margin for write operations.

Persons skilled in the art will recognize that many variations of the teachings of this invention can be practiced that still fall within the claims of this invention which follow.

High Performance Bus Interface -62-

-62-